import React, { useState, useEffect } from 'react';

import { Copy, Volume2, RefreshCw, Globe, Sparkles } from 'lucide-react';

import { Button } from '@/components/ui/button';

import { Textarea } from '@/components/ui/textarea';

import { Select, SelectContent, SelectItem, SelectTrigger, SelectValue } from '@/components/ui/select';

import { Card, CardContent } from '@/components/ui/card';

import { Badge } from '@/components/ui/badge';

import { useToast } from '@/hooks/use-toast';

const Index = () => {

const [sourceText, setSourceText] = useState('');

const [translatedText, setTranslatedText] = useState('');

const [sourceLanguage, setSourceLanguage] = useState('en');

const [targetLanguage, setTargetLanguage] = useState('es');

const [isTranslating, setIsTranslating] = useState(false);

const [apiKey, setApiKey] = useState('');

const { toast } = useToast();

const languages = [

{ code: 'en', name: 'English', flag: '🇺🇸' },

{ code: 'es', name: 'Spanish', flag: '🇪🇸' },

{ code: 'fr', name: 'French', flag: '🇫🇷' },

{ code: 'de', name: 'German', flag: '🇩🇪' },

{ code: 'it', name: 'Italian', flag: '🇮🇹' },

{ code: 'pt', name: 'Portuguese', flag: '🇵🇹' },

{ code: 'ru', name: 'Russian', flag: '🇷🇺' },

{ code: 'ja', name: 'Japanese', flag: '🇯🇵' },

{ code: 'ko', name: 'Korean', flag: '🇰🇷' },

{ code: 'zh', name: 'Chinese', flag: '🇨🇳' },

{ code: 'ar', name: 'Arabic', flag: '🇸🇦' },

{ code: 'hi', name: 'Hindi', flag: '🇮🇳' },

{ code: 'tr', name: 'Turkish', flag: '🇹🇷' },

{ code: 'nl', name: 'Dutch', flag: '🇳🇱' },

{ code: 'sv', name: 'Swedish', flag: '🇸🇪' },

];

const translateText = async (text: string) => {

if (!text.trim() || !apiKey.trim()) return;

setIsTranslating(true);

// Simulated translation API call - replace with actual API

try {

await new Promise(resolve => setTimeout(resolve, 1000)); // Simulate API delay

// Mock translation for demo purposes

const mockTranslations = {

'Hello world': { es: 'Hola mundo', fr: 'Bonjour le monde', de: 'Hallo Welt' },

'How are you?': { es: '¿Cómo estás?', fr: 'Comment allez-vous?', de: 'Wie geht es dir?' },

'Good morning': { es: 'Buenos días', fr: 'Bonjour', de: 'Guten Morgen' },

};

const translation = mockTranslations[text as keyof typeof mockTranslations]?.[targetLanguage as keyof typeof mockTranslations['Hello world']]

|| [${targetLanguage.toUpperCase()}] ${text};

setTranslatedText(translation);

} catch (error) {

toast({

title: "Translation Error",

description: "Failed to translate text. Please check your API key.",

variant: "destructive",

});

} finally {

setIsTranslating(false);

}

};

useEffect(() => {

const timeoutId = setTimeout(() => {

if (sourceText && apiKey) {

translateText(sourceText);

}

}, 500);

return () => clearTimeout(timeoutId);

}, [sourceText, sourceLanguage, targetLanguage, apiKey]);

const copyToClipboard = (text: string) => {

navigator.clipboard.writeText(text);

toast({

title: "Copied!",

description: "Text copied to clipboard",

});

};

const swapLanguages = () => {

setSourceLanguage(targetLanguage);

setTargetLanguage(sourceLanguage);

setSourceText(translatedText);

setTranslatedText(sourceText);

};

return (

<div className="min-h-screen bg-gradient-to-br from-purple-900 via-blue-900 to-indigo-900 relative overflow-hidden">

{/\* Background decoration \*/}

<div className="absolute inset-0 opacity-20" style={{

backgroundImage: url("data:image/svg+xml,%3Csvg width='60' height='60' viewBox='0 0 60 60' xmlns='http://www.w3.org/2000/svg'%3E%3Cg fill='none' fill-rule='evenodd'%3E%3Cg fill='%239C92AC' fill-opacity='0.05'%3E%3Ccircle cx='30' cy='30' r='4'/%3E%3C/g%3E%3C/g%3E%3C/svg%3E")

}}></div>

<div className="container mx-auto px-4 py-8 relative z-10">

{/\* Header \*/}

<div className="text-center mb-12">

<div className="flex items-center justify-center gap-3 mb-4">

<div className="relative">

<Globe className="h-12 w-12 text-white" />

<Sparkles className="h-6 w-6 text-yellow-400 absolute -top-1 -right-1 animate-pulse" />

</div>

</div>

<h1 className="text-5xl font-bold text-white mb-4 bg-gradient-to-r from-blue-400 to-purple-400 bg-clip-text text-transparent">

AI Translation Studio

</h1>

<p className="text-xl text-blue-200 max-w-2xl mx-auto">

Break language barriers with real-time AI-powered translation. Supporting 15+ languages for global communication.

</p>

</div>

{/\* API Key Input \*/}

{!apiKey && (

<Card className="max-w-md mx-auto mb-8 bg-white/10 backdrop-blur-md border-white/20">

<CardContent className="p-6">

<div className="text-center mb-4">

<h3 className="text-white font-semibold mb-2">Enter Your API Key</h3>

<p className="text-blue-200 text-sm">Enter any text as API key for demo purposes</p>

</div>

<div className="flex gap-2">

<input

type="password"

placeholder="Enter API key..."

value={apiKey}

onChange={(e) => setApiKey(e.target.value)}

className="flex-1 px-3 py-2 bg-white/20 border border-white/30 rounded-lg text-white placeholder:text-blue-200 focus:outline-none focus:ring-2 focus:ring-blue-400"

/>

<Button onClick={() => setApiKey(apiKey || 'demo-key')} variant="secondary">

Start

</Button>

</div>

</CardContent>

</Card>

)}

{apiKey && (

<div className="max-w-6xl mx-auto">

{/\* Language Selection \*/}

<div className="flex items-center justify-center gap-4 mb-8">

<div className="flex items-center gap-2">

<span className="text-white font-medium">From:</span>

<Select value={sourceLanguage} onValueChange={setSourceLanguage}>

<SelectTrigger className="w-48 bg-white/10 border-white/20 text-white">

<SelectValue />

</SelectTrigger>

<SelectContent>

{languages.map((lang) => (

<SelectItem key={lang.code} value={lang.code}>

<span className="flex items-center gap-2">

<span>{lang.flag}</span>

<span>{lang.name}</span>

</span>

</SelectItem>

))}

</SelectContent>

</Select>

</div>

<Button

variant="ghost"

size="icon"

onClick={swapLanguages}

className="text-white hover:bg-white/10 hover:rotate-180 transition-all duration-300"

>

<RefreshCw className="h-5 w-5" />

</Button>

<div className="flex items-center gap-2">

<span className="text-white font-medium">To:</span>

<Select value={targetLanguage} onValueChange={setTargetLanguage}>

<SelectTrigger className="w-48 bg-white/10 border-white/20 text-white">

<SelectValue />

</SelectTrigger>

<SelectContent>

{languages.map((lang) => (

<SelectItem key={lang.code} value={lang.code}>

<span className="flex items-center gap-2">

<span>{lang.flag}</span>

<span>{lang.name}</span>

</span>

</SelectItem>

))}

</SelectContent>

</Select>

</div>

</div>

{/\* Translation Interface \*/}

<div className="grid md:grid-cols-2 gap-6">

{/\* Source Text \*/}

<Card className="bg-white/10 backdrop-blur-md border-white/20 hover:bg-white/15 transition-all duration-300">

<CardContent className="p-6">

<div className="flex items-center justify-between mb-4">

<div className="flex items-center gap-2">

<span className="text-lg">

{languages.find(l => l.code === sourceLanguage)?.flag}

</span>

<span className="text-white font-medium">

{languages.find(l => l.code === sourceLanguage)?.name}

</span>

</div>

<Badge variant="secondary" className="bg-blue-500/20 text-blue-200">

Source

</Badge>

</div>

<Textarea

value={sourceText}

onChange={(e) => setSourceText(e.target.value)}

placeholder="Type your text here..."

className="min-h-[200px] bg-transparent border-white/20 text-white placeholder:text-blue-200 focus:border-blue-400 resize-none"

/>

<div className="flex items-center justify-between mt-4">

<span className="text-blue-200 text-sm">

{sourceText.length} characters

</span>

<div className="flex gap-2">

<Button

variant="ghost"

size="icon"

onClick={() => copyToClipboard(sourceText)}

className="text-white hover:bg-white/10"

disabled={!sourceText}

>

<Copy className="h-4 w-4" />

</Button>

<Button

variant="ghost"

size="icon"

className="text-white hover:bg-white/10"

disabled={!sourceText}

>

<Volume2 className="h-4 w-4" />

</Button>

</div>

</div>

</CardContent>

</Card>

{/\* Translated Text \*/}

<Card className="bg-white/10 backdrop-blur-md border-white/20 hover:bg-white/15 transition-all duration-300">

<CardContent className="p-6">

<div className="flex items-center justify-between mb-4">

<div className="flex items-center gap-2">

<span className="text-lg">

{languages.find(l => l.code === targetLanguage)?.flag}

</span>

<span className="text-white font-medium">

{languages.find(l => l.code === targetLanguage)?.name}

</span>

</div>

<Badge variant="secondary" className="bg-green-500/20 text-green-200">

Translation

</Badge>

</div>

<div className="min-h-[200px] bg-white/5 border border-white/20 rounded-lg p-4 text-white">

{isTranslating ? (

<div className="flex items-center gap-2 text-blue-200">

<div className="animate-spin rounded-full h-4 w-4 border-b-2 border-blue-400"></div>

Translating...

</div>

) : translatedText ? (

<p>{translatedText}</p>

) : (

<p className="text-blue-200 italic">Translation will appear here...</p>

)}

</div>

<div className="flex items-center justify-between mt-4">

<span className="text-blue-200 text-sm">

{translatedText.length} characters

</span>

<div className="flex gap-2">

<Button

variant="ghost"

size="icon"

onClick={() => copyToClipboard(translatedText)}

className="text-white hover:bg-white/10"

disabled={!translatedText}

>

<Copy className="h-4 w-4" />

</Button>

<Button

variant="ghost"

size="icon"

className="text-white hover:bg-white/10"

disabled={!translatedText}

>

<Volume2 className="h-4 w-4" />

</Button>

</div>

</div>

</CardContent>

</Card>

</div>

{/\* Features \*/}

<div className="grid md:grid-cols-3 gap-6 mt-12">

<Card className="bg-white/5 backdrop-blur-md border-white/20 hover:bg-white/10 transition-all duration-300">

<CardContent className="p-6 text-center">

<div className="w-12 h-12 bg-blue-500/20 rounded-full flex items-center justify-center mx-auto mb-4">

<Globe className="h-6 w-6 text-blue-400" />

</div>

<h3 className="text-white font-semibold mb-2">15+ Languages</h3>

<p className="text-blue-200 text-sm">Support for major world languages including English, Spanish, French, German, and more.</p>

</CardContent>

</Card>

<Card className="bg-white/5 backdrop-blur-md border-white/20 hover:bg-white/10 transition-all duration-300">

<CardContent className="p-6 text-center">

<div className="w-12 h-12 bg-purple-500/20 rounded-full flex items-center justify-center mx-auto mb-4">

<Sparkles className="h-6 w-6 text-purple-400" />

</div>

<h3 className="text-white font-semibold mb-2">Real-time Translation</h3>

<p className="text-blue-200 text-sm">Get instant translations as you type with advanced AI-powered language processing.</p>

</CardContent>

</Card>

<Card className="bg-white/5 backdrop-blur-md border-white/20 hover:bg-white/10 transition-all duration-300">

<CardContent className="p-6 text-center">

<div className="w-12 h-12 bg-green-500/20 rounded-full flex items-center justify-center mx-auto mb-4">

<Volume2 className="h-6 w-6 text-green-400" />

</div>

<h3 className="text-white font-semibold mb-2">Voice Support</h3>

<p className="text-blue-200 text-sm">Listen to pronunciations and speak your text for hands-free translation experience.</p>

</CardContent>

</Card>

</div>

</div>

)}

</div>

</div>

);

};

export default Index;